FACTA UNIVERSITATIS (NIŠ)
SER.: ELEC. ENERG. vol. 19, no. 2, August 2006, 261-269

Competitive Learning Algorithms for Data Clustering

Georgeta Budura, Corina Botoca, and Nicolae Miclău

Abstract: This paper presents and discusses some competitive learning algorithms for data clustering. A new competitive learning algorithm, named the dynamically penalized rival competitive learning algorithm (DPRCL), is introduced and studied. It is a variant of the rival penalized competitive algorithm [1] and it performs appropriate clustering without knowing the clusters number, by automatically driving the extra seed points far away from the input data set. It does not have the "dead units" problem. Simulations results, performed in different conditions, are presented showing that the performance of the new DPRCL algorithm is better comparative with other competitive algorithms.

Keywords: Competitive learning algorithms, radial basis function neural networks

1 Introduction

Competitive learning is an efficient tool for data clustering, widely applied in a variety of signal processing problems such as data compression [2], classification [3, 4], adaptive noise cancellation [1, 5, 6] image retrieval [7] and image processing [8–10].

The typical competitive learning algorithm, named *k*-means algorithm [11], partitions the input data set into *k* categories (called clusters) each finally represented by its centre, that adaptively changes, starting from some initial values named seed points The algorithm computes the squared distance between the input vector and the centres, chooses the winning centre, the one having the minimum distance, and moves it closer to the input vector. The major deficiency of the *k*-means algorithm is that it needs to know the exact number of clusters *k*, before

Manuscript received June 28, 2005

G. Budura and C. Botoca are with "Politehnica" University of Timisoara, Faculty of Electronics and Telecommunications, Communications Department, V. Parvan 2, Timisoara, Romania (e-mail: [georgeta.budura, corina.botoca, nicolae.miclau]@etc.utt.ro).

performing data clustering. Otherwise, it will lead to a poor clustering performance. Unfortunately, it is often hard to determine k in advance in many practical problems. Other problems are that the classification depends on the initials centres values of the RBF, on the type of the chosen distance, on the number of classes. The k-means algorithm has also the ”dead units” problem, which means that if a centre is inappropriately chosen, it may never be updated, thus it may never represent a class.

During the last fifteen years there have been developed new advanced algorithms that eliminate the ”dead units” problem and perform clustering without predeciding the exact cluster number, as for example: the frequency competitive algorithm (FSCL) [4], the incremental k-means algorithm, the rival penalizing competitive algorithm (RPCL) [12].

The frequency sensitive competitive learning is an extension of the k-means algorithm, that circumvent the ”dead units” problem [4] by introducing a parameter, named the relative winning frequency or ”conscience”. The centres chance to win the competition is directly proportional with the relative winning frequency. The FSCL algorithm reduces the learning rate of the frequent winners, so their chance to win the competition. Some successful applications of the FSCL algorithm are feature extraction [13] and image compression [14]. Although the FSCL algorithm can almost successfully assign one or more seed points to a cluster without the ”dead units” problem, it also needs knowing the exact number of clusters.

Another algorithm which can perform clustering without knowing the clusters number is a variant of the k-means algorithm, the k-means incremental algorithm. It gradually increases the number of clusters under the control of a threshold parameter, which is however difficult to be decided.

The rival penalized competitive learning algorithm [12] performs appropriate clustering without knowing the clusters number and also it eliminates the ”dead units” problem. The RPCL algorithm rewards the winning center and penalizes with a de-learning rate the second winner, named rival. The algorithm is quite simple and provides a better convergence than the k-means and the FSCL algorithms. Although the RPCL algorithm had success in some applications, such as nonlinear channel equalization [1], color image segmentation [8], images features extraction [10], it is rather sensitive to the selection of the de-learning rate.

To eliminate this drawback, in this paper it is introduced a new competitive algorithm, the dynamically penalized rival competitive learning algorithm. The DRPCL algorithm circumvents the selection problem of the de-learning rate by always fixing it at the same value as the learning rate and dynamically controlling it. If the first rival distance to the winner is closer than the one between the winner and the input, the rival will be more penalized; otherwise the penalization will be

gradually attenuated as the distance between the winner and the rival increases. This idea is consistent with the social scenario in our daily life. For example, the electoral competition between two candidates always will become more intense if their performances are closer. The DRPCL algorithm drives away the extra number of seed points much faster than the RPCL algorithm.

2 Competitive Learning Algorithms

2.1 The k-means algorithm

The standard k-means algorithm [11] calculates the distance between the input vector and the centres vector. The distance may be of different types, but usually the Euclidian norm is used:

$$\|x(n) - c_i(n)\| = \sqrt{[x_1(n) - c_{1i}(n)]^2 + \cdots + [x_m(n) - c_{mi}(n)]^2}, \tag{1}$$

where $x(n)$ is the input vector, c_i is the centre vector i, m is the vectors dimension and N is the number of the centres. The centre j with a minimum distance is declared winner:

$$j = \arg\min \|x(n) - c_i(n)\| \quad i = 1, \ldots, N. \tag{2}$$

The winning centre is moved with a fraction η towards the input

$$c_i(n+1) = c_i(n) - \eta\,[x(n) - c_i(n)]\,. \tag{3}$$

The learning rate η may be constant or descendant with a fraction, for example:

$$\eta(n+1) = \eta(n) - \frac{1}{N}, \tag{4}$$

where N represents the centres number.

The weights are randomly initialized, usually at the input vector values. Equations (2) and (3) are than applied iteratively until the algorithm converges or freezes as the number of iterations reaches a specified value, respectively the descending learning rate becomes zero or a very small value.

The classical k-means algorithm has the ”dead units” problem. That is, if a center is initialized far away from the input data set in comparison with other centers, it may never win the competition, so it may never update, becoming a dead unit.

2.2 The frequency sensitive competitive algorithm

To solve the "dead units" problem it has been introduced the so called "frequency sensitive competitive learning" algorithm [4] or competitive algorithm "with conscience". Each centre counts the number of times when it has won the competition and reduces its learning rate consequently. If a center has won too often "it feels guilty" and it pulls itself out of the competition. The FSCL algorithm is an extension of *k*-means algorithm, obtained by modifying relation (2) according to the following one:

$$j = \arg\min \gamma_i \|x(n) - c_i(n)\| \quad i = 1, \ldots, N, \tag{5}$$

with the relative winning frequency γ_i of the centre c_i defined as

$$\gamma_i = \frac{s_i}{\sum_{i=1}^{N} s_i}, \tag{6}$$

where s_i is the number of times when the centre c_i was declared winner in the past. So the centers that have won the competition during the past have a reduced chance to win again, proportional with their frequency term γ. After selecting out the winner, the FSCL algorithm updates the winner with equation (3) in the same way as the *k*-means algorithm, and meanwhile adjusting the corresponding s_i with the following relation:

$$s_i(n+1) = s_i(n) + 1. \tag{7}$$

The FSCL algorithm can almost always successfully distribute the N centres into the input data set without the "dead units problem", but only when the clusters number is known.

2.3 The rival penalized competitive learning algorithm

The rival penalized competitive algorithm [1] performs appropriate clustering without knowing the clusters number. It determines not only the winning centre j but also the second winning center r, named rival

$$r = \arg\min \gamma_i \|x(n) - c_i(n)\|, \quad i = 1, \ldots, N, \quad i \neq j. \tag{8}$$

The second winning centre will move away its centre from the input with a ratio β, called the de-learning rate. All the other centres vectors will not change. So the learning law can be synthesized in the following relation:

$$c_i(n+1) = \begin{cases} c_i(n) + \eta\,[x(n) - c_i(n)] & \text{if} \quad i = j \\ c_i(n) - \beta\,[x(n) - c_i(n)] & \text{if} \quad i = r \\ c_i(n) & \text{if} \quad i = j \quad \text{and} \quad i \neq r \end{cases} \tag{9}$$

If the learning speed η is chosen much greater than β, with at least one order of magnitude, the number of the output data classes will be automatically found. In other words, suppose that the number of classes is unknown and the centres number N is greater than the clusters number, than the centres vectors will converge towards the centroids of the input data classes. The RPCL will move away the rival, in each iteration, converging much faster than the k-means and the FSCL algorithms. The number of extra seed points, respectively the difference between N and the number k of classes will be driven away from the data set. If N is smaller than the number of input data classes, than the network will oscillate during training, indicating that the clusters number must be increased.

2.4 The dynamically penalized rival competitive learning algorithm

The DRPCL is a variant of the RPCL algorithm. It determines not only the winning neuron, but also the second winning neuron, the first rival. Comparative to the RPCL algorithm, the DRPCL algorithm applies a new mechanism to dynamically control the rival penalization. For this it introduces a new term, the penalization strength:

$$p(c_i(n)) = \frac{\min(\|x(n) - c_w\|, \|c_w - c_r\|)}{\|c_w - c_r\|}, \tag{10}$$

where c_w and c_r are the centre of the winner, respectively of the rival. With this factor the de-learning rate β in equation (9) becomes:

$$\beta = \eta \, p(c_i(n)). \tag{11}$$

It can be noticed that the value of p(ci) in relation (10) is always between 0 and 1, which can be therefore regarded as the probability of rival penalization. If the condition $\|x(n) - c_w\| \geq \|c_w - c_r\|$ is satisfied, than the rival will be fully penalized with the learning rate η. Otherwise, the rival will be penalized with the de-learning rate η p(ci(n), which is gradually attenuated as the distance between the winner and its rival increases. So, the DRPCL algorithm is actually a generalization of the RPCL algorithm, which moves away the undesired centres much faster than the RPCL algorithm, because its de-learning rate is greater.

3 Simulations Results

A complex radial basis function neural network (RBF-NN) [1], [2], was trained with the FSCL, RPCL and the new DRPCL algorithm, using complex input data. Different numbers of clusters and numbers of seed points were used in our experiments. The input data were generated using a white Gaussian noise, around the

desired centres clusters, independently for the real part from the imaginary part. Several experiments were done for different noise dispersions σ^2. The RBF-NN centres were randomly initialized to a subset of the input data. The learning rate was chosen to η=0.001 for all the three algorithms. The de-learning rate for the RPCL algorithm was β=0.0001.

In all cases the new DRPCL algorithm could find the desired centres clusters, had driven away the extra number of seed points and had a faster convergence than the other two algorithms. It was noticed that when DRPCL gives an inappropriate clustering at current time step, i.e., there were two or more seed points located in one true cluster, the rival penalization rate was not less than 0.25η in average if the data were uniformly distributed in the cluster. Such a big de-learning rate could however lead the RPCL algorithm to the total break down. Hence, the de-learning rate in the equation (11) was generally much smaller than 0.25η, resulting that the DRPCL algorithm drove away the extra seed points far away from the clusters, much faster than the RPCL .

Example 3.1

There were used 600 complex input data generated using a white Gaussian noise around three points:(1; j) , (1; 5j)and (5; 5j). The input data formed three clusters as it can be seen in Figure 1(a), Figure 1(b) and Figure 1(c). The six initial positions of the RBF-NN centres were randomly located at (0.2580; 0.2849j), (1.4659; 5.1359j), (0.3893; 5.3331j), (5.2045; 5.1298j), (1.9193; 5.4489) and (5.5869; 5.1937j);

Figures 1(a), 1(b) and 1(c) represent the desired states, the input noisy states $x(n)$, the initial and final positions of the RBF centres states c(n), in the case of a dispersion σ^2=0.36 after 100 training epochs using the FSCL, the RPCL and the DRPCL algorithms. The FSCL algorithm failed to find the desired centres. Both other algorithms, RPCL and DRPCL, succeed to orientate the RBF centres to the desired states.

Comparing Figure 1(b) with Figure 1(c) it can be observed that the DRPCL algorithm has driven away the extra number of seed points much faster than the RPCL algorithm and it has found closer positions of RBF centres to the desired states than the RPCL algorithm, so its convergence is better.

Example 3.2

There were used 1600 noisy input states x(n) with a dispersion of σ^2=0.1, around 16 desired complex points. The RBF-NN had 20 initial centres points, as it can be seen in Figures 2(a), 2(b) and 2(c). Figures 2(a), 2(b) and 2(c) represent the simula-

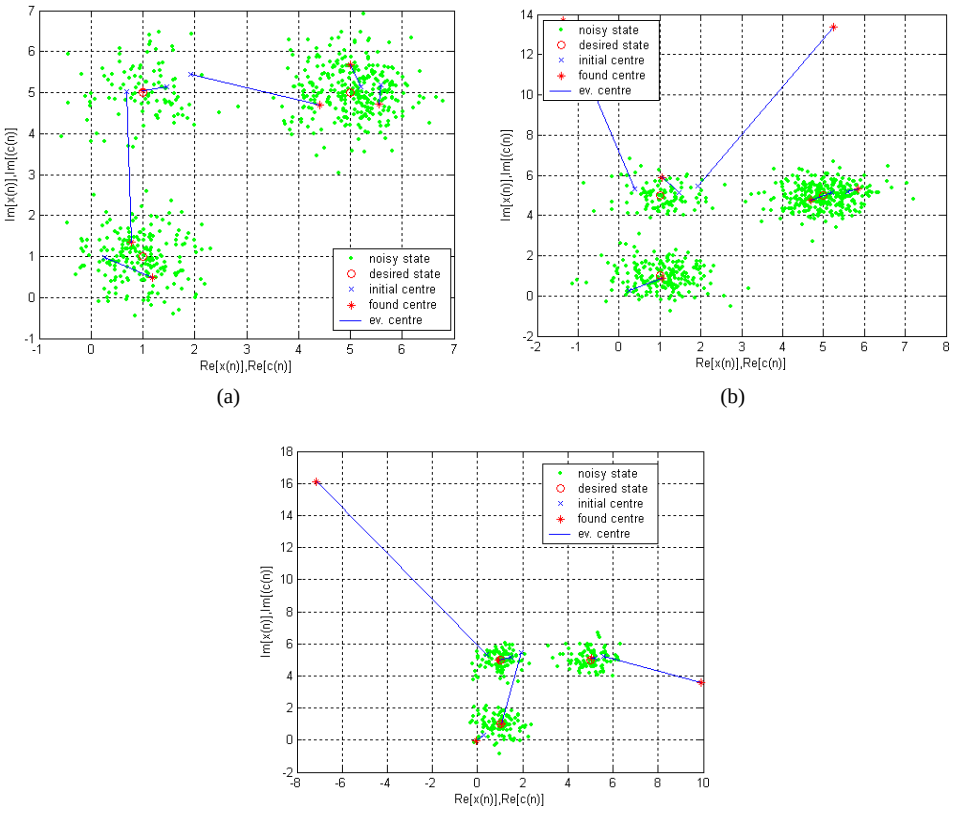

Fig. 1. (a) The desired states, the input noisy states $x(n)$, the initial and final positions of the RBF centres in case of a noise dispersion σ^2=0.36 after 100 training epochs, using the FSCL algorithm. (b) The desired states, the input noisy states $x(n)$, the initial and final positions of the RBF centres, in case of a noise dispersion σ^2=0.36 after 100 training epochs, using the RPCL algorithm. (c) The desired states, the input noisy states $x(n)$, the initial and final positions of the RBF centres, in case of a noise dispersion σ^2=0.36 after 100 training epochs, using the DRPCL algorithm.

tions results after 30 training epochs, for the FSCL, RPCL and DRPCL algorithms.

It can be seen that the FSCL algorithm failed to find the desired because it cannot deal with the extra number of seed points. centres. Both other algorithms, RPCL and DRPCL, succeed to orientate the RBF centres to the desired states. In addition, the RPCL and the DRPCL algorithm have driven away the extra number of seed points. As one can observe the DRPCL algorithm has driven the extra number of seed points much faster away and could find closer positions of RBF centres to the desired states, than the RPCL algorithm, so its convergence is better.

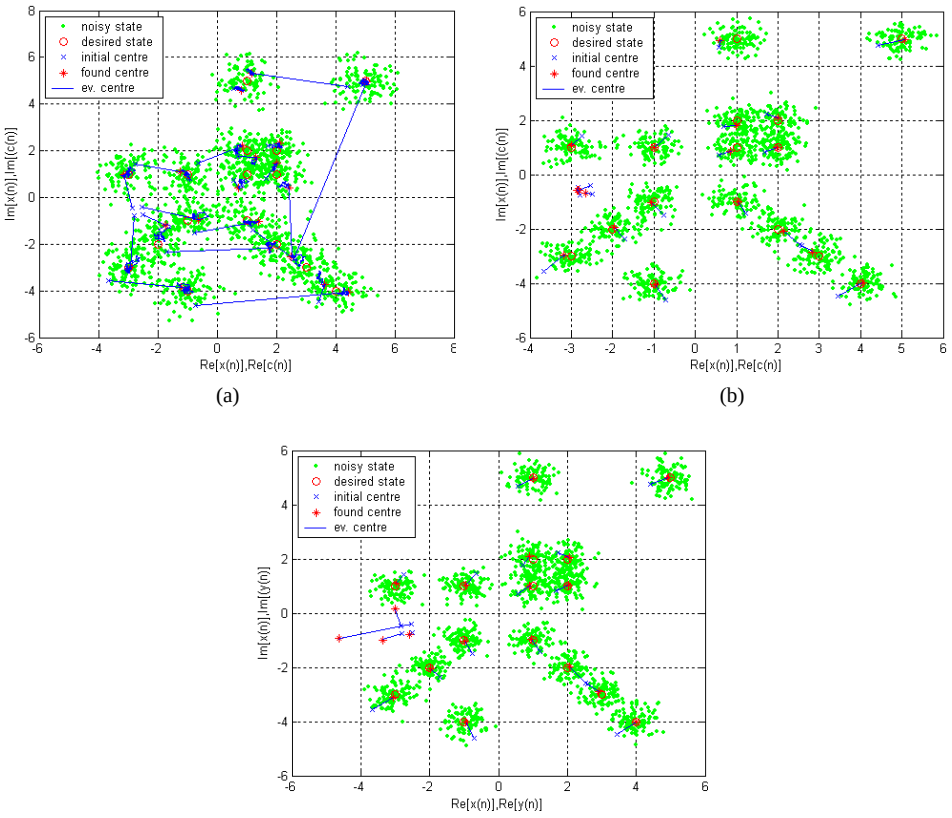

Fig. 2. (a) The input desired states, the corrupted signal $x(n)$, the initial and final positions of the RBF centres in case of a noise dispersion σ^2=0.1, after 30 training epochs, using the FSCL algorithm. (b) The input desired states, the corrupted signal $x(n)$, the initial and final positions of the RBF centres in case of a noise dispersion σ^2=0.1, after 30 training epochs, using the RPCL algorithm. (c) The input desired states, the corrupted signal $x(n)$, the initial and final positions of the RBF centres in case of a noise dispersion σ^2=0.1, after 30 training epochs, using the DRPCL algorithm.

4 Conclusions

The proposed competitive algorithm, namely DRPCL, while training the centres of the RBF network, rewards the winner and dynamically penalizes its closest rival, inverse proportional with the distance between the winner and its rival. In comparison with the classic *k*-means algorithm, it doesn't have the "dead units" problem. If compared with the FSCL algorithm it doesn't need to know the clusters number. Comparative to the RPCL algorithm, it has a better and faster convergence. So the DRPCL algorithm is adequate to the adaptive clustering of fast varying signals corrupted by noise.

References

[1] N. Miclau, C. Botoca, and G. Budura, "Nonlinear complex channel equalization using a radial basis function neural network," in *Proc. of NEUREL 2004*, Belgrade, Serbia and Montenegro, 2004, pp. 73–78.

[2] T. Hofmann and J. M. Buhmann, "Competitive learning algorithms for robust vector quantization," *IEEE Trans. on Signal Processing*, vol. 46, no. 6, pp. 1665–1675, 1998.

[3] R. Gray, "Vector quantization," *IEEE ASSP Magazine*, pp. 4–49, Apr. 1984.

[4] S. C. Ahalt, A. K. Krishnamurty, P. Chen, and D. E. Melton, "Competitive algorithms for vector quantization," *Neural Networks*, no. 3, pp. 277–291, 1990.

[5] S. Bouchired and M. Ibnkahla, "Decision neuronale appliquee a l'egalisation de canaux satellitaires mobiles," in *Proceedings of Gretsi'99*, Vannes, France, Sept. 13–17, 1999, pp. 1125–1128.

[6] X. Wang, H. Liu, J. Lu, and T. Yohagy, "Combining recurrent neural networks with self-organizing maps for channel equalization," *IEEE Trans. on Communications*, vol. E85-E, pp. 2227–2235, 2002.

[7] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik, "Blobworld: A system for region-based image indexing and retrieval," in *Proc. of Third Int. Conf. On Visual Information and Information Systems*, Amsterdam, 1999, pp. 509–516.

[8] L. T. Law and Y. M. Cheung, "Color image segmentation using rival penalized controlled competitive learning," in *Proc. of 2003 International Joint Conference on Neural Networks (IJCNN'2003)*, Portland, Oregon, USA, July 20-24, 2003.

[9] V. Ramos and F. Muge, "Map segmentation by colour cube genetic k-means clustering," in *Lecture Notes in Computer Science*. Springer-Verlag, 2000, vol. 1923.

[10] T. Nakamura, K. Terada, A. Shibata, J. Morimoto, H. Adachi, and H. Takeda, "Development of a cheap on-board vision mobile robot for robotic soccer research," in *Proc. of International Conference on Intelligent Robots and System*, Victoria, BC, Canada, Oct. 1998, pp. 431–436.

[11] Hecht-Nielsen, *Neurocomputing*. New York: Addison-Wesley Publishing Company, 1990.

[12] L. Xu, A. Krzyzak, and A. E. Oja, "Rival penalized competitive learning for clustering analysis. rbf net and curve detection," *IEEE Trans. on Neural Networks*, no. 4, pp. 636–64, 1993.

[13] H. C. Card and S. Kamasu, "Competitive learning and vector quantization in digital vlsi," *Neurocomputing*, vol. 18, pp. 195–227, jan 1998.

[14] C. H. Chang, P. Xu, R.Xiao, and T. Srikanthan, "New adaptive quantization method based on self-organization," *IEEE Trans. on Neural Networks*, vol. 16, pp. 237–249, Jan. 2005.